2006

Group key agreement protocols with implicit key authentication

Jisoo Kim

MS Project Proposal

Group Key Agreement Protocols with Implicit Key Authentication

Jisoo Kim
Rochester Institute of Technology
Department of Computer Science
jsk4445@cs.rit.edu

Feburary 7, 2005

MS Project Committee

Chairman
Stanislaw P. Radziszowski

Reader
Unknown

Observer
Unknown

Abstract

There have been a lot of studies performed on secure group communication over unsecured channels such as the Internet and Ad Hoc network. Most of the results are focused on cryptographic methods to share secret keys within the group. In the real world, however, we cannot establish an application for group communication without considering authentication of each peer (group member) since the adversary could digitally disguise itself and intrude into the key sharing process without valid membership. Therefore, authentication is an inevitable component for any other secure communication protocols as well as peer group communication.

On the classical design of group key protocols, each peer should be authenticated by a separate and centralized authentication server (e.g. Kerberos). Although many practical protocols present efficient ways for authentication, we are still facing the necessity of optimization between authentication and group key distribution. In that sense, implicit key authentication is an ideal property for group key protocols since, once it is possibly put into practice, we don't need any separate authentication procedure as a requisite.

There was an attempt to devise implicit key authentication service in conjunction with group key protocol; Authenticated Group Diffie-Hellman (A-GDH) and its stronger version (SA-GDH). Unfortunately, both were proved to have some weakness from the man-in-the-middle attack. In the forthcoming project, fixes for A-GDH and SA-GDH using a Message Authentication Code (MAC) scheme will be proposed and performance evaluation will be carried out from implementation and experimentation for each; A-GDH, SA-GDH, A-GDH with MAC, and SA-GDH with MAC.

Contents

1 Introduction

Since 2-party Diffie-Hellman key exchange was first proposed in 1976 [3], its contributory nature has attracted many cryptographers into trying to extend it to a group setting. Among those efforts, Group Diffie-Hellman (GDH) in [8] is thought as one of the successful extensions of Diffie-Hellman to the n-party case. There are several versions of GDH, among which GDH.2 and GDH.3 are considered as practical group protocols (see the details in [8]). Nevertheless, GDH cannot stand alone, as other group key distribution protocols, since authentication of each peer (group member) should precede the group key sharing procedure in a practical application. Although there are useful authentication techniques for group communication protocols, most of them depend upon a centralized server, *trusted third party*. This not only increases communication costs but also deteriorates security of the protocol.

Implicit Key Authentication, which will be discussed in the next section, is an ideal property for secure group communication since it gets rid of the need for a separate authentication machanism during key sharing. In [1] and [2], Ateniese *et al.* proposed authenticated versions of GDH, which include Implicit Key Authentication; Authenticated GDH.2 (A-GDH.2) and Strong Authenticated GDH.2 (SA-GDH.2). Several years after A-GDH.2 and SA-GDH.2 were proposed, Pereira et al. [7] proved that the authenticated versions of GDH still have the weakness from man-in-the-middle attack (2-party Diffie-Hellman has the same flaw).

In this project, I will propose fixes for A-GDH and SA-GDH using a Message Authentication Code (MAC) scheme, which will protect the messages exchanged from adversary's forgery. Among the practical MAC protocols, which are all on the trade-off between efficiency and reliability, I will mainly consider efficiency of the protocol since group communication is usually performed synchronously and whatever the adversary does should be executed in relatively tiny time. Finally, there will be performance evaluation from implementation and experimentation for each; A-GDH, SA-GDH, A-GDH with MAC, and SA-GDH with MAC.

2 Notation and Properties

In this section, I will present the notation which will be used in the forthcoming project and discuss the properties of authenticated GDH protocols.

2.1 Notation for Authenticated GDH protocols

The notation which will be used in this project is mostly taken from [1] and [2], and all arithmetic operations are performed in a cyclic group G of prime order q which is a subgroup of Z_p^* for a prime p such that $p = kq + 1$ for some small $k \in N$ (e.g. $k = 2$). The notation used in this project is shown in Table 1.

n	Number of group members
i, j	Indices of group members
M_i	i-th group member; $i \in [1, n]$
p, q	p, q prime, $q \mid \phi(p)$
G	Unique subgroup of Z_p^* of order q
α	Exponentiation base; generator of G
x_i	Long-term secret key of M_i
r_i	M_i's secret exponent (nounce) $\in Z_q$
S_n	Group key shared among n memebers
$S_n(M_i)$	M_i's view on a group key
K_{ij}	Long-term secret shared by M_i and M_j
$F()$	A function from G to Z_q
I_{ij}	Intermediate value computed by M_i for M_j

Table 1: Notation

2.2 Properties of Authenticated GDH protocols

A-GDH.2 and SA-GDH.2 support commonly desired properties for group key distribution protocols; Perfect Forward Secrecy (PFS), Resistance to Known-Key Attacks, Key Authentication, and so on. [1] and [2] present the definitions of these properties and the proofs that our protocols satisfy those. In addition, A-GDH.2 and SA-GDH.2 possess characteristics as the following:

- Key Agreement protocol
- Contributory protocol
- Provide Implicit Key authentication

The characteristics above distinguish authenticated versions of GDH from other group key distribution protocols. The definitions for the above are given below. (These are adapted from [1] or [2])

Definition 1 *A **key agreement protocol** is a key establishment technique whereby a shared secret key is derived by two or more specified parties as a function of information contirbuted by, or associated with, each of these, such that no party can predetermine the resulting value.*

Definition 2 *A key agreement protocol is **contirbutory** if each party equally contributes to the key and guarantees its freshness.*

We can see that the basic two-party Diffie-Hellman is a contributory as well as a key agreement protocol. However, A-GDH and SA-GDH may have a contraversy in their contributory property according to what the word *equally* means. Practically in group protocols, the meaning of *equally* is regarded as contributing each member's public key to the key distribution process in an even way.

Definition 3 *Let $\mathcal{R}$ be an n-party key agreement protocol, $\mathcal{M}$ be the set of protocol parties and let S_n be a secret key jointly generated as a result of $\mathcal{R}$. We say that $\mathcal{R}$ provides* ***implicit key authentication*** *if each $M_i \in \mathcal{M}$ is assured that no party $M_q \notin \mathcal{M}$ can learn the key S_n unless aided by a dishonest $M_j \in \mathcal{M}$.*

[1] and [2] present the proof that A-GDH.2 and SA-GDH.2 provide implicit key authentication. As mentioned above, this property removes the need for a separate authentication process.

3 Authenticated Group Diffie-Hellman protocols

In this section, I will discuss A-GDH.2, SA-GDH.2, the attacks on them, and the fixed using MAC. The discussion will be mainly on *Initial Key Agreement*[1] operations since this project is focused on the authentication when the group communication is initiated. Note that authentication during member joining or group merging depends on the policies, related to the problem of *who authenticate(s) whom*. The detail cases under topological changes of dynamic group are given in [9].

3.1 A-GDH.2 protocol

A-GDH.2 is an authenticated version of GDH.2. The protocol description of A-GDH.2 is given in [1] and [2].

As we see, the upflow stage (*i.e.* the protocol rounds from M_1 to M_{n-1}) is identical to GDH.2. The only difference is that, on the broadcast stage, the group controller (the last member M_n) sends the intermediate value I_{ni} to each member M_i ($i \in [1, n-1]$), hiding with the long-term secret key between M_i and M_n, $K_{in} = F(\alpha^{x_i x_n}(mod\ q))$. Since x_i and x_n refer to the long-term secret keys of M_i and M_n, K_{in} can be computed only by M_i and M_n. The problem to find α^{ab} from given α^a and α^b is known as a computational Diffie-Hellman problem[2]. The last round of this protocol guarantees that no one without an appropriate long-term key could compute the same group key. For example, M_i receives $\alpha^{\frac{r_1 \cdots r_n}{r_i} \cdot K_{in}}$ from the group controller M_n. M_i can compute the group key $S_n = \alpha^{r_1 \cdots r_n}$ only if M_i can compute K_{in}^{-1} where $K_{in} \cdot K_{in}^{-1} = 1(mod\ q)$. Therefore, $M_I \notin G$ disguised as M_i cannot compute the group key without knowing x_i, the long-term secret key of M_i. This satisfies the property of implicit key authentication.

However, A-GDH.2 is a relatively weak form of implicit key authentication since the authentication is performed only between M_i and M_n, $i \in [1, n-1]$. An example of A-GDH.2 with four parties is given in Figure 1. If we take a snapshot in the view of M_3, we can see that M_3 receives the sequence of intermediate values $\{\alpha^{r_2}, \alpha^{r_1}, \alpha^{r_1 r_2}\} = \{I_{21}, I_{22}, I_{23}\}$ from M_2 and sends $\{\alpha^{r_2 r_3}, \alpha^{r_1 r_3}, \alpha^{r_1 r_2}, \alpha^{r_1 r_2 r_3}\} = \{I_{31}, I_{32}, I_{33}, I_{34}\}$ to M_4.

[1] *Initial key agreement* refers to the very first group key agreement. This operation takes place at the time of group *genesis*. In the meanwhile, *Auxiliary Key Agreement* refers to all subsequent key agreement operations through membership changes after the initial group key is once established. (See the details in [9])

[2] It is known that a computational Diffie-Hellman problem is very hard to solve since there is no efficient way to find the solution [10]. The group protocols based on Diffie-Hellman rely on this hardness.

Note that there is no authentication until the last memeber M_n broadcasts $I_{ni} = \alpha^{\frac{r_1 \cdots r_n}{r_i} \cdot K_{ni}}$ to each member M_i.

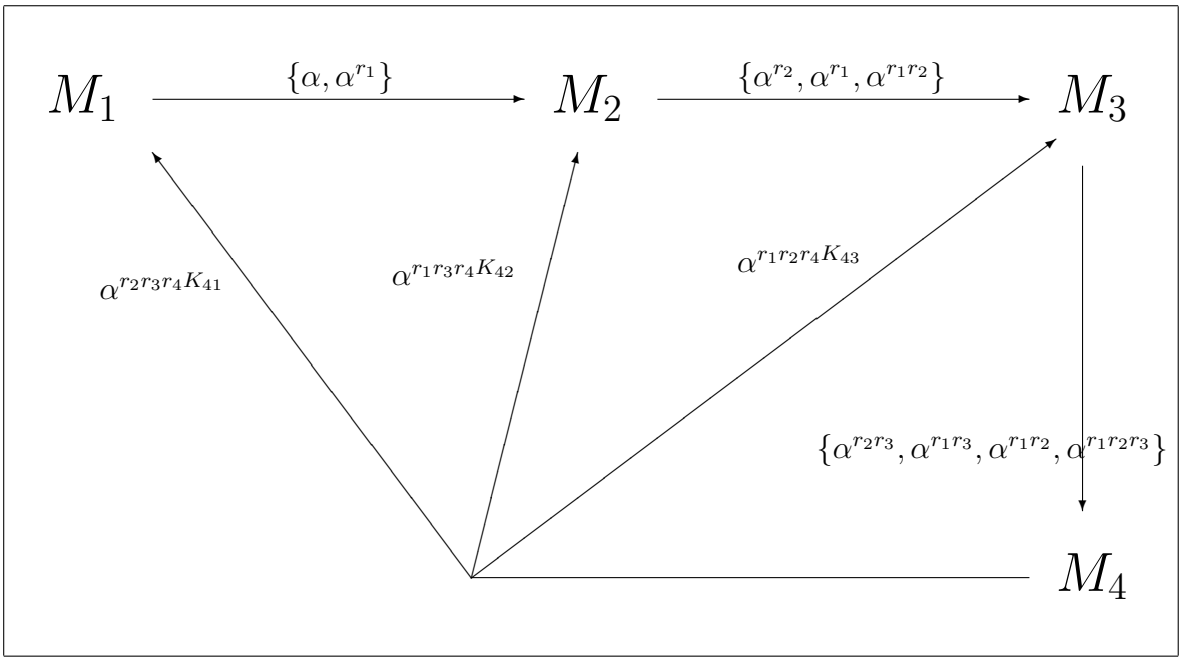

Figure 1: An example of A-GDH.2 with 4 members

3.2 SA-GDH.2 protocol

As mentioned above, A-GDH.2 is not strong enough in a certain situation since authenticating function is centralized to the last member (group controller). In order to provide stronger authentication service, Ateniese *et al* also designed SA-GDH.2 in which every member authenticates the other members (see [1] and [2] for the protocol details).

Unlike A-GDH.2, SA-GDH.2 requires each intermediate value[3] for M_i, given by M_k, to be computed by using K_{ki} $(0 < i \neq k \leq n)$ from the upflow stage as well so that each member can not only equally contribute to authentication but also be explicitly aware of the exact group membership when the protocol is initiated. Another advatage of SA-GDH.2 over A-GDH.2 is that each member performs the same sequence of computational steps and the same number of exponentiations. An example of SA-GDH.2 with four members is given in Figure 2. Obviously, SA-GDH.2 is more expensive in computation of intermediate values than A-GDH.2.

Although SA-GDH.2 has higher cost of exponentiation than A-GDH.2, if the group settings (*e.g.* group membership, communication order of each member, etc) are determined in advance of the protocol's beginning (this seems to be a common situation in group communication), the computational costs can be mostly saved by pre-computation.

[3]In SA-GDH.2 protocol, different number of intermediate values computed by M_i are sent to the next member M_{i+1}. For example, M_{i+1} in SA-GDH.2 receives from M_i a sequence of intermediate values $\{I_{i1}, I_{i2}, \ldots I_{in}\}$, instead of $\{I_{i1}, I_{i2}, \ldots I_{i(i+1)}\}$ in A-GDH.2.

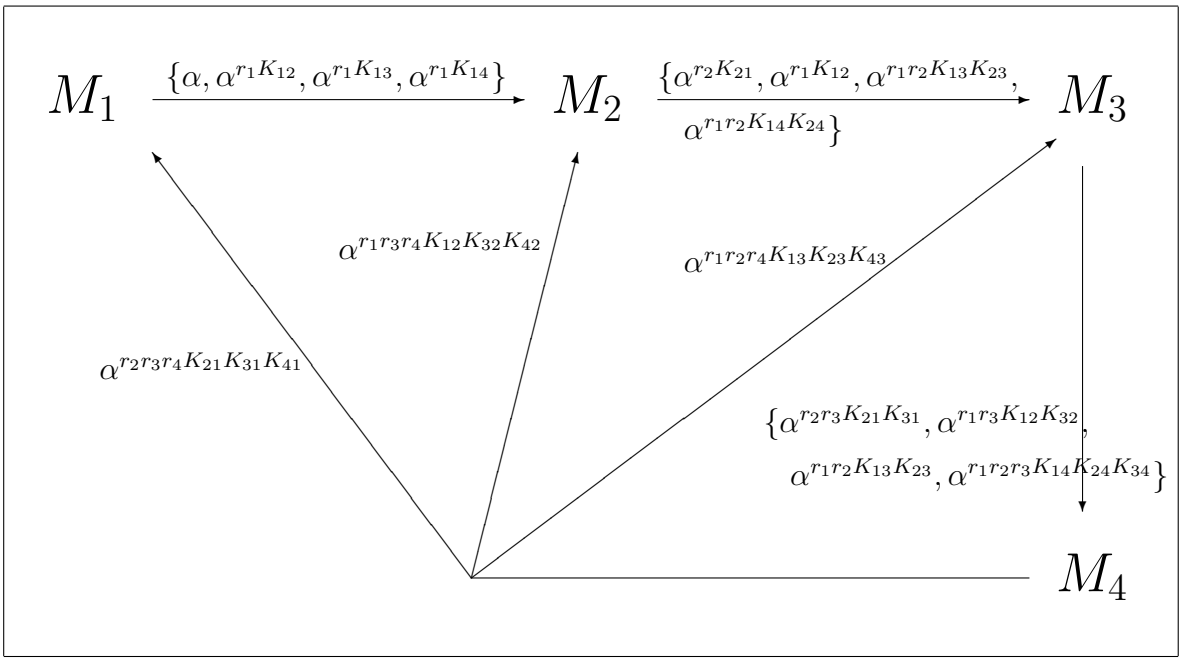

Figure 2: An example of SA-GDH.2 with 4 members

Let I_{ij} be the intermediate value for M_j computed by M_i, then M_i computes I_{ij} as (1).

$$I_{ij} = \begin{cases} {I_{(i-1)j}}^{E_{ij}} & \text{if } i \neq j \\ I_{(i-1)j} & \text{if } i = j \end{cases} \tag{1}$$

$$E_{ij} = r_i \cdot K_{ij} \pmod{q} \tag{2}$$

As you can see in (2), the exponential value E_{ij} can be pre-computed at the preliminary stage of key agreement protocol. Therefore, we can minimize the additional computation on SA-GDH.2 in a practical way.

3.3 Attacks on authenticated GDH protocols

Pereira *et al.* provide security analyses on authenticated GDH protocols. First, they tried to find security holes of A-GDH.2 in conjuction with the major secure properties in [5]: Implicit Key Authentication, Perfect Forward Secrecy, and Resistance to known-keys attacks. However, their attacks have some impractical conditions on the intruder's ability which includes the followings:

- An intruder can eavesdrop the intended message sent by any participant in the current group session.
- An intruder can immediately forge the intended message sent by any participant without being detected.

Although both conditions above seems to be more infeasible as group communication gets bigger in the size and more dynamic with the membership, this shows the relative

vulnerability of A-GDH.2. The similar technique and condition can be applied to attack SA-GDH.2 (See the details in [6]). The effect from the attack on SA-GDH.2 is more limited than on A-GDH.2. However, SA-GDH.2 still has flaws on its security as a group communication protocol. Finally, Pereira *et al.* proposed the proof on the generic insecurity of authenticated GDH protocols in 2004 [7]. They proved that it is impossible to design a scalable authenticated group key agreement protocol based on the same building blocks as the authenticated GDH ones under the conditions on the intruders described above.

One of the solutions given by Pereira *et al.* in the conclusion of [7] is the use of message authentication codes (MAC) which prevents the intruders from forging messages. They thought this separates the key generation part of the protocol from the authentication mechanisms. However, I think that we should not necessarily separate those two parts and that we can design an authenticated GDH protocol with implicit key authentication using MAC, which has slightly heavier computation than the original design of A-GDH and SA-GDH.

3.4 Fixes for A-GDH.2 and SA-GDH.2 using a MAC

Against the active adversary who can forge messages in the network, MAC has been proved to be a reliable technique and widely used in the real world. For example, Secure Socket Layer (SSL) protocol, a standard of a secure protocol supported by many web browsers, uses message authentication codes (MAC) in order to authenticate both the source of a message and its integrity wihout the use of any additional mechanisms.

Among many MAC protocols being used in communication applications, the Keyed-Hash Message Authentication Code (HMAC) is a cryptographic standard adopted by National Institute of Standards and Technology (NIST) and proven to be secure as long as the underlying hash function[4] has some reasonable cryptographic strengths.

HMACs require two functionally distinct parameters, a message input and a secret key[5] known only to the message sender and intended recipient. Since the authenticated GDH protocols, both A-GDH.2 and SA-GDH.2, have a secret key K_{ik} shared between M_i and M_k, we can design the fix with MAC without any separate authentication mechanisms. The fixes with MAC are shown in Figure 3. Note that I_{ik} is the intermediate value for M_k computed by M_i and that the sequence of intermediate values in the up-flow stages (the rounds from 1 to $n-1$) is different between A-GDH.2 and SA-GDH.2[6]. Finally, each member can verify the intermediate values with MAC tags. If the MAC tags are not valid, the receiver may request the sender (the former member in the protocol sequence) to send the message again.

The length of MAC tag is not fixed in the HMAC algorithm. Since this project is focused on the computational efficiency, the minimum security level is enough here. According to [4], the length of the tag, t, shall be at least $\frac{L}{2}$ bytes where L is the block

[4]SHA1 and MD5 have become standard hash functions for many cryptographic applications.

[5]In order that HMAC should work, the message sender and the receiver must share a pre-arranged secret key. See the details in [4].

[6]In the case with four members, for example, the third member M_3 receives $\{I_{21}, I_{22}, I_{23}\}$ in A-GDH.2 and $\{I_{21}, I_{22}, I_{23}, I_{24}\}$ in SA-GDH.2.

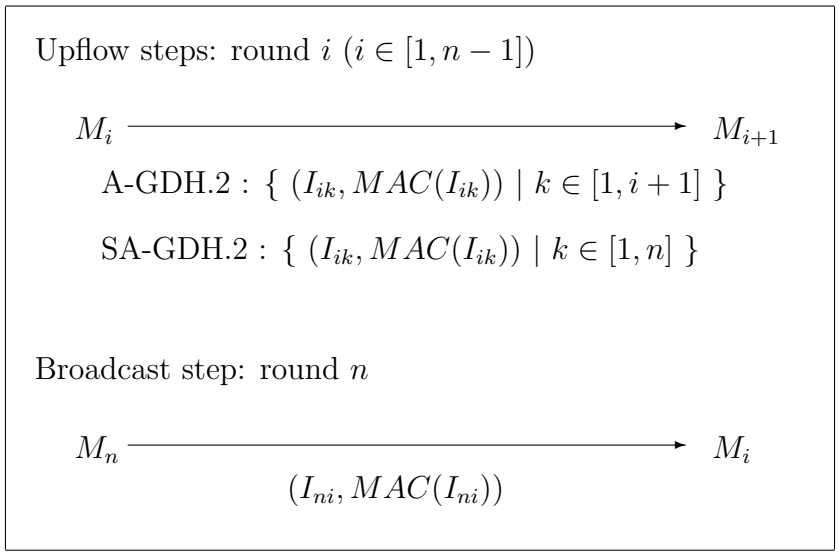

Figure 3: Using MAC in authenticated GDH protocols

size of the output of the hash function[7]. Therefore, we can take $\frac{L}{2}$ as t. Even though the intruder attempts *birthday attack*, a well-known method to attack MAC, the synchronous nature[8] of group communication hinders him (or her) from carrying out his (or her) plan or the attack is easily detected by MAC.

4 Deliverables

The goal of this project is to probe how practical the authenticated versions of GDH and their fixes with MAC are. Therefore, the forthcoming project should include the followings:

- Project Report

  Theoretical review on the work of authenticated GDH protocols, HMAC, and related studies .

  Defining the authentication policy (*who authenticate(s) whom*) for member joining/leaving and group merging/disjoing, and then designing an authenticated protocol for dynamic group membership extended from authenticated GDH.

- Experiment

  Experiments for computaitional performance evaluation of A-GDH.2, SA-GDH.2, A-GDH.2 with HMAC, and SA-GDH.2 with HMAC, in the initial key agreement process. When the group communication commences with arbitrary number of members, those protocols have different complexity of computations. I expect to

[7] $\frac{L}{2} \leq t \leq L$

[8] Authenticated GDH protocols also support asynchronous communication. In this case, however, the implicit key authentication service is not as useful as in the synchronous communication since we can take the classical design (separate authentication mechanism) without implicit key authentication.

show how badly or gradually those compuations degrade as each protocol is tested in order of low computation.

- Source Code

References

[1] G. Ateniese, M. Steiner, and G. Tsudik. *Authenticated group key agreement and friends*, in Proceedings of the 5th ACM Conference on Computer And Communications Securit*my*, pages 17-26, San Francisco, USA, 1998. ACM Press.

[2] G. Ateniese, M. Steiner, and G. Tsudik. *New multi-party authentication services and key agreement protocols*. IEEE Journal on Selected Areas in Communication, 2000.

[3] W. Diffie, M. Hellman. *New Directions In Cryptography*. IEEE Transactions on Information Theory, IT-22(6):644-654, November 1976.

[4] National Institute of Standards and Technology, *The Keyed-hash Message Authentication Code (HMAC)*, Federal Information processing Standards Publication 198, 6 March 2002.

[5] O. Pereira and J.J. Quisquater. *A security analysis of the Cliques protocols suites*, in Proceedings of the 14th IEEE Computer security Foundations Workshop - CSFW'01, pages 73-81, Cap Breton, Canada, 2001. IEEE Computer Society Press.

[6] O. Pereira and J.J. Quisquater. *Some attacks upon authenticated group key agreement protocols*, Journal of Computer Security, 11(4):555-580, 2003.

[7] O. Pereira and J.J. Quisquater. *Generic Insecurity of Cliques-type Authenticated Group Key Agreement Protocols*, Computer Security Foundations Workshop, 2004. Proceedings. 17th IEEE, 2004.

[8] M. Steiner, G Tsudik, and M. Waidner. *Diffie-Hellman key distribution extended to groups*, in Third ACMConference on Computer and Communications Security. Mar, 1996, pp.31-37, ACM Press.

[9] M. Steiner, G. Tsudik, and M. Waidner. *CLIQUES: A new approach to group key agreement*, in IEEE International Conference on Distributed Computing Systems, May 1998.

[10] Douglas R. Stinson. *Cryptography: Theory and Practice, Second Edition*. CRC Press Series on discrete mathematics and its applications. CRC Press, 2002. ISBN 1-58488-206-9. [Page 265]